МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(МОСКОВСКИЙ ПОЛИТЕХ)**

09.03.01 Информатика и вычислительная техника

Образовательная программа (профиль)

«Интеграция и программирование в САПР»

Кафедра «СМАРТ технологии»

Методическое пособие

по дисциплине:

**Проектная деятельность**

на тему: Создание анимации сборки и разборки МКПП

Куратор: / Толстиков Антон Витальевич /

*подпись ФИО, уч. звание и степень*

Студент: /Муравьёв Андрей Геннадьевич 191-325/

*подпись ФИО, группа*

Москва, 2020 г.

**Цель работы:** Создать скрипт для управления камерой

**Задачи:**

1. Изучить построение скриптов для Unity
2. Написать скрипт управления камерой

**Методы:**

Работа с Unity 3D.

Для создания скрипта в Unity необходимо в области “Project” перейти во вкладку “Assets” и далее во вкладку “Scripts”. После чего нажать ПКМ в этой области и выбрать последовательно вкладки “Create” и “C# Script” (Рисунок 1).

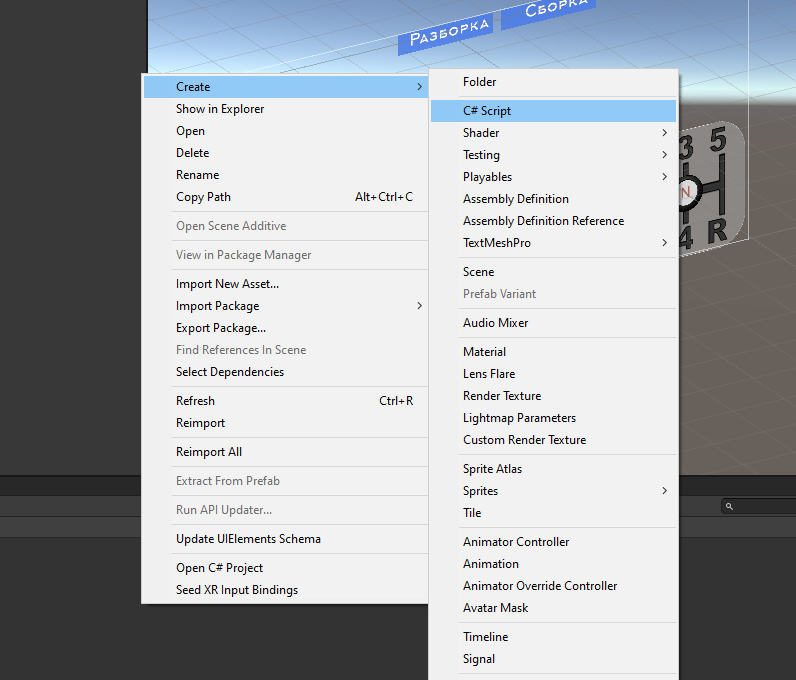


Рисунок 1 – создание пустого скрипта

Нажимаем ЛКМ по скрипту дважды, чтобы открыть его в установленной IDE, в моем случае – Microsoft Visual Studio (Рисунок 2).

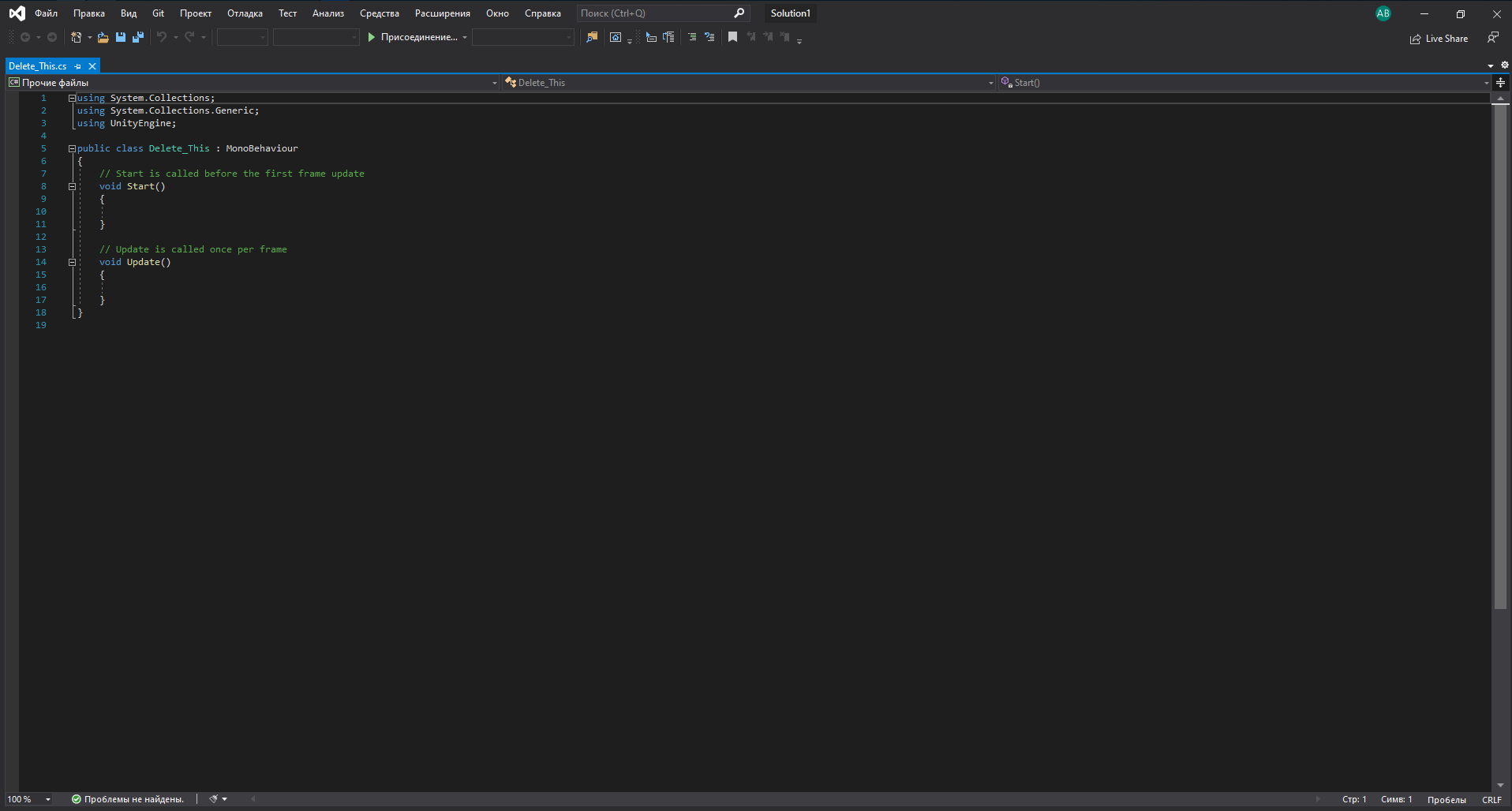


Рисунок 2 – открытый скрипт в IDE

Для проекта необходимы такие скрипты как, например: скрипт управления камерой, скрипт запуска анимации сборки/разборки, скрипт для переключения между сценами.

Рассмотрим скрипт управления камерой.

Мы должны создать переменные, которые будут отвечать за чувствительность мыши, скорость передвижения и вектор направления. Для этого в глобальной области создаем их (Рисунок 3).

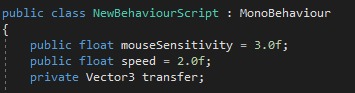


Рисунок 3 – переменные

Кроме них в глобальной области нужны переменные, отвечающие за максимальные углы поворота камеры по осям Х и У. Для этого записываем это переменные. Так же необходимо объявить использование Quaternion(Обеспечивает вращение). Получаем следующий код (Рисунок 4).

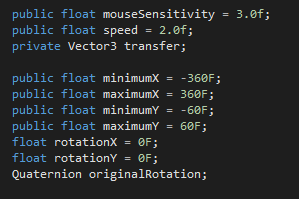


Рисунок 4 – объявление переменных

Далее идет метод Start(),в котором мы запишем начало управления, поэтому мы пока что его пропускаем. После него идет метод Update(), в котором мы должны записать свой код для управления.

Мы должны создать условие, при котором будет происходить вращение камерой, в нашем случае – когда зажата ЛКМ и LeftShift. В теле созданного условия к объявленной переменной rotationX мы прибавляем положение мыши на оси X, умноженное на чувствительность мыши, получаем следующую строку:

rotationX += Input.GetAxis("Mouse X") \* mouseSensitivity;

Аналогичным образом пишем строку для rotationY.

После этого переменным нужно задать угол, для этого пишем следующую строку:

rotationX = ClampAngle(rotationX, minimumX, maximumX);

Аналогичным образом пишем строку для rotationY.

Далее используем Quaternion для самого вращения камерой. Записываем следующие строки:

Quaternion xQuaternion = Quaternion.AngleAxis(rotationX, Vector3.up);

Quaternion yQuaternion = Quaternion.AngleAxis(rotationY, Vector3.left);

После этого для самого вращения камерой пишем строку:

transform.rotation = originalRotation \* xQuaternion \* yQuaternion;

В результате получаем следующий код (Рисунок 5).

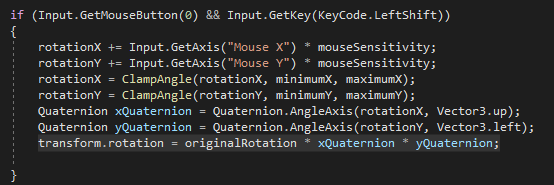


Рисунок 5 – код условия поворота

После этого пишем строки для изменения позиции камеры. В итоге метод Update() должен выглядеть следующим образом (Рисунок 6).

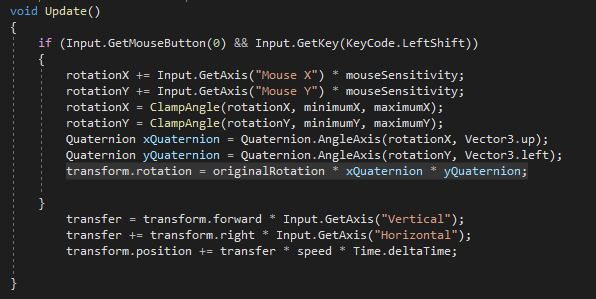


Рисунок 6 – метод Update()

Далее нужно написать функцию, выполняющую условия перехода на +360 градусов. Нужно это для того, чтобы вращение камерой было постоянным, а не заканчивалось на 360 градусах (Рисунок 7).

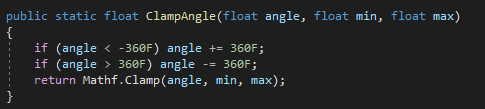


Рисунок 7 – условия постоянного вращения камеры

После этого в раннее упомянутый метод Start() мы пишем запуск Quaternion. Метод Start () должен выглядеть следующим образом (Рисунок 8).

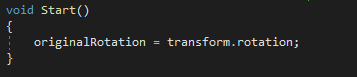


Рисунок 8 – метод Start()

Весь код скрипта указан на рисунке 9.

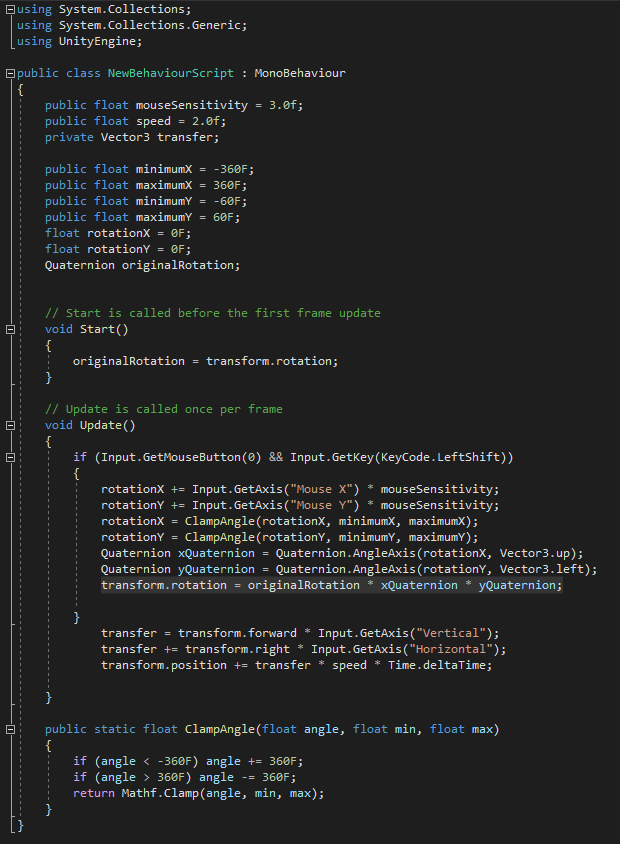


Рисунок 9 – код скрипта управления камерой

Далее нам нужно наш скрипт поместить на саму сцену, а именно на объект Main Camera. Для этого зажав ЛКМ на нашем скрипте в области “Project” мы перетаскиваем его на объект Main Camera, который расположен в дереве (Рисунок 10).

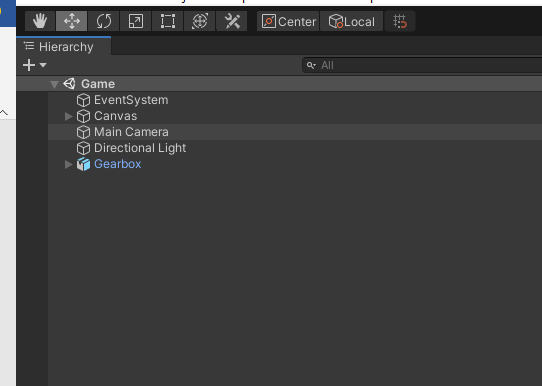


Рисунок 10 – объект Main Camera в дереве

Проверяем работоспособность скрипта запустив проект.